CS 2316

# Homework 9a – Login

**Due: Wednesday, March 28th, 2011 before 11:55 PM**
**Out of 100 points**

---

**Files to submit:** **1. HW9.py**

**This is an INDIVIDUAL assignment!**
Collaboration at a reasonable level will not result in substantially similar code. Students may only collaborate with fellow students currently taking CS 2316, the TA's and the lecturer. Collaboration means talking through problems, assisting with debugging, explaining a concept, etc. You should not exchange code or write code for others.

For Help:
- TA Helpdesk – Schedule posted on class website.
- Email TA's or use T-Square Forums

Notes:
- **Don't forget to include the required comments and collaboration statement (as outlined on the course syllabus).**
- ***Do not wait until the last minute*** *to do this assignment in case you run into* problems.
- **Read the entire specifications document before starting this assignment.**

---

# Premise

This homework is the first of a two part assignment in which you will do some real world applications with databases and GUIs. For this first part of the assignment, you will be building a two page GUI that offers the user the chance to login, or in the case that the user does not have an account yet, register and then log in. You'll find this quite applicable to building programs with user-accounts, and your future CS4400 project option.

Although this program does not require knowing much new information, it does test simple SQL query and insertion abilities as well as new coding practices to create two GUI pages that can be opened and closed at the click of a button. **This new information may take some understanding, so please do not wait until the last minute to begin this assignment.**

# Database Format Information

The database for this part of the assignment only has 1 table of users, conveniently called User.

```
CREATE TABLE User
        (Username       VARCHAR(25)     PRIMARY KEY,
        FirstName       VARCHAR(20),
        Password        VARCHAR(10)     NOT NULL)
```

The code above was used to create the table. As can be seen, the Username and Password are non-NULL variables, but FirstName can be left blank and NULL will assigned instead of a name. Note that the Username and FirstName are different, a Username is required, but a FirstName is not.

# Multiple GUI Pages

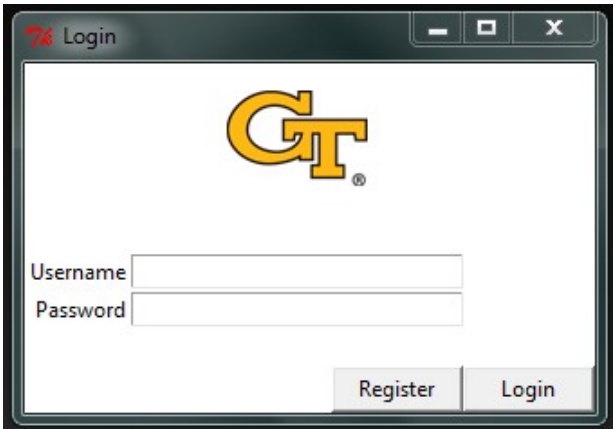
Multiple GUI pages may not seem difficult, but it takes a certain method to create new windows while destroying old ones. It will involve the creation and button execution of several helper methods to transition between pages. Your __init__ method, for instance, should not actually create a GUI page, but instead call a method to do so. There will be two methods for GUI creations, two pages for writing and retrieving user information, and several helper methods for transitioning. It's up to you as to how many transition methods you will need, but basically one per transition should be enough. In these transition methods, you will .destroy() a certain Tk() instance created for a certain page, while calling a method that creates a different Tk() instance for a different page. This means that you will be using two different Tk() instances defined within your class and more specifically defined and created in your GUI creation methods. To do this you use your 'TkinterInstance=Tk()' and 'TkinterInstance.mainloop()' methods all within your same GUI creation method.

## Method Name: **__init__**

This method will only be used to call your first GUI creation method for the login page.

## Method Name**: LoginPage**

This method is called to create a new instance of your GUI class. This method is responsible for arranging and initializing your GUI. You should create a GUI which looks like the one below.



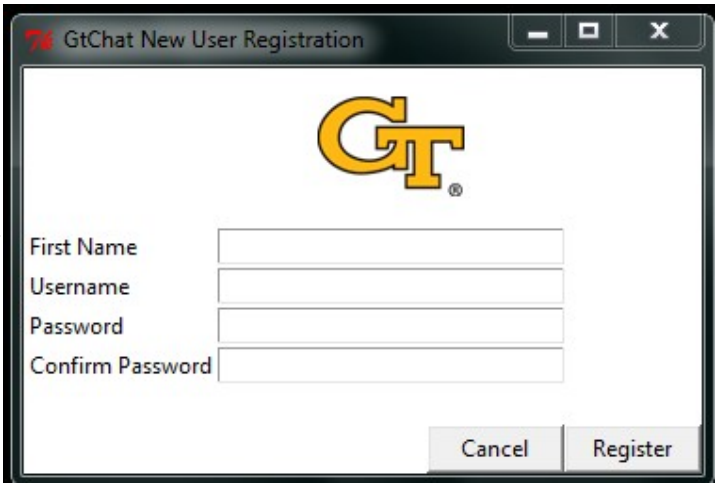
You may generate this window however you wish, but it is *highly* suggested that you use the grid layout manager. Things to note about the GUI you see:

- The top image can be any logo that is Gt related. You can find an image anywhere online that allows webscraping and use the image url for urllib.request.urlretrieve(). Alternatively you may use an image from your own computer and upload that to t-square when you submit your code. You should use a small .gif image and the method(s) described here:
  http://effbot.org/tkinterbook/photoimage.htm
  http://www.summet.com/dmsi/html/guiProgramming.html#adding-images-to-your-gui

- The labels are sticky to the east and the entry boxes are width 30 and state normal
- Clicking on the "Register" button should call the Register method.
- Clicking on the "Login" should call the LoginCheck method.

## Method Name: **Register**

This method will simply create a new GUI instance for filling out registration information. The GUI should look something like this:




You may generate this window however you wish, but it is *highly* suggested that you use the grid layout manager. Things to note about the GUI you see:

- The top image is the same as the one in the login page.
- Each entry is width 30 and state normal.
- The labels are all sticky to the west.
- Clicking on the "Cancel" button calls a helper method to .destroy() this GUI instance and recall the LoginPage method.
- Clicking on the "Register" button calls the RegisterNew method.

## Method Name: **Connect**

Parameters:

None

Return Value:

Database connection

Description:

This method will use a try/except statement to attempt to connect to the database. If it cannot connect, have it display a message box asking the user to check their Internet connection. This method returns the database connection object. The connection parameters are the username and password we provided you.

## Method Name: **RegisterNew**

Parameters:

None

Return Value:

None

Description:

This method will take the entries from the register page and put them into the database. You must check that both passwords entered are the same and that the username and password entries are not left blank. Use message boxes to describe any problems with the way the register page was filled out. Also check to see if the username is already used within the database. If so, use a message box to express this. If everything is ok, you can insert into the database. (Hint: The First Name entry can be left blank and in that case would not be inserted into the database.) Once registered, a message box should tell the user that they are now registered and a helper method should be called to .destroy() this GUI instance and recall the LoginPage. Be sure to .commit() the database when you're done.

## Method Name: **LoginCheck**

Parameters:

None

Return Value:

None

Description:

This method will check to see if what was typed into the login page matches any users currently in the database. To do this you call the Connect method, create a cursor from the database connection that is returned, and check what was passed to see if any matches are returned. You may also want to get the first name of the user to use on the 2nd part of his homework later (hint hint). If it finds no matches, it should return a message box explaining that the user entered an unrecognizable username/password combination. If it does match, you should remember the username for use later in part 2 of the homework, and then have it display a message box saying you logged in successfully. (You should also close the login GUI, as you are finished with it.)

A sample gtChat login that may already exist in the database is username "bboyer6" and password "raptor". You can try using this to test your login code before

you have your registration code working. (Please do not mess up this account information, or other students will not be able to successfully use it for testing.)

## Grading:

You will earn points as follows for each piece of functionality that works correctly according to the specifications.

| | | |
|---|---|---|
| **LoginPage GUI** | | **20** |
| GUI has image at top | 6 | |
| GUI has all components with proper characteristics | 4 | |
| Login works as described | 4 | |
| Register works as described | 4 | |
| GUI uses message boxes for errors | 2 | |
| **RegisterPage GUI** | | **20** |
| GUI has image at top | 6 | |
| GUI has all components with proper characteristics | 4 | |
| Register works as described | 4 | |
| Cancel brings user back to login page | 4 | |
| GUI uses message boxes for errors | 2 | |
| **LoginCheck()** | | **25** |
| Uses Connect() properly | 5 | |
| Correctly checks to find matching user | 10 | |
| Uses message boxes for errors with user match | 4 | |
| Closes GUI with match | 4 | |
| Commits db when done | 2 | |
| **RegisterNew()** | | **25** |
| Uses Connect() properly | 5 | |
| Correctly puts user in database | 10 | |
| Uses message boxes for errors with user entries | 4 | |
| Checks for usernames already taken | 4 | |
| Commits db when done | 2 | |
| **Connect()** | | **10** |
| Correctly connects to database and returns database | 10 | |